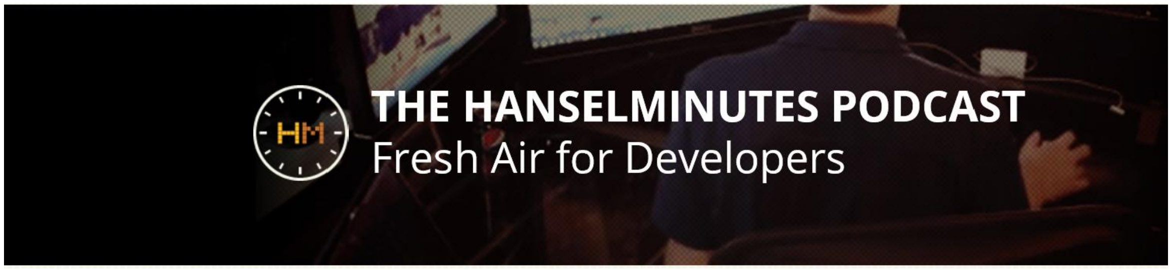

Text transcript of show #363

March 22, 2013

HTML5, JavaScript, Chrome and the Web Platform with Paul Irish

Scott sits down with Chrome Developer Advocate Paul Irish to talk HTML5, JavaScript, Chrome and the Web. What Chrome Developer Tools features make web dev easier? While WebKit marches on, should we embrace or fear monoculture? Will modules make JavaScript apps easier to write? Where does Windows fit into the world of web development?

Our Sponsors

telerik
deliver more than expected

http://www.telerik.com

Lawrence Ryan: From www.hanselminutes.com, it's Hanselminutes, a weekly discussion with web developer and technologist, Scott Hanselman. This is Lawrence Ryan announcing show #363, recorded live Thursday, March 14, 2013. Support for Hanselminutes is provided by Telerik, offering the best in developer tools and support, online at www.telerik.com. And by Franklins.Net, makers of GesturePak, a powerful gesture recording and recognition system for Microsoft Kinect for Windows developers. Details at www.gesturepak.com. In this episode, Scott talks with Paul Irish about HTML5, JavaScript, Chrome, and the Web platform.

Scott Hanselman: Hi, this is Scott Hanselman. This is another episode of Hanselminutes, and today we've got Paul Irish. How are you, sir?

Paul Irish: Hello. Doing good. Doing great.

Scott Hanselman: Very cool. Thanks for coming on to talk to me. So you work at Google…you are in Chrome Developer Relations?

Paul Irish: Yeah, that's absolutely right. Developer advocate on Chrome. I'm all about…these days, I'm all about developer productivity, making sure that developers are as productive as they can be when building web apps that are amazing on desktop, on mobile. And I'm also…you know, people develop with best practices that are performant (sic). Things like that.

Scott Hanselman: So I think…I love that word 'advocate.' I think that you and I kind of have similar jobs, but we also work for these giant companies where people assume that we have to represent all the company's interests. So it's like, "You work for Google. You must care really deeply about ads."

Paul Irish: (laughter)

Scott Hanselman: But from watching you, you care about the Web, and people think I care about Xbox or whatever. Or I got yelled at recently for not having a Windows Phone. It's like…you know? Kind of not my thing.

Paul Irish: Yeah, absolutely. My passion is that I want the Web to win. I'm really passionate about it as an application platform, and I think there's still a lot we can do, but I am excited about the state that it is right now. So that's my focus, and it's not a Google point of view, per se. There's a lot of things that Google does that could be even better when it comes towards the Web and its promise. I'm very happy to be honest about those kinds of things.

Scott Hanselman: Do you feel like you're kind of in a pretty sweet situation in the sense of you've got this big company behind you that lets you push the Web forward, kind of regardless if it helps Google or not? The Web will win. The Web is winning. The avalanche has begun.

Paul Irish: Yeah. It's really fantastic to be somewhere where I can have this sort of reach and impact that I think really benefits the situation. That's like when a startup is like, "Hey, Paul. You want to go lead a front-end engineering team?" And I'm like, "Well, I do, partly want to go just hack 100% on Web stuff," but also I think we're not done yet from a Web advocacy standpoint. So I think it's great to be here.

Scott Hanselman: Yeah. Just when I think we are even close to done, I'm like, "No." We've been at 49% for the last 10 years.

Paul Irish: (laughter) Yeah.

Scott Hanselman: The progress bar is not moving. Although Chrome definitely changed the speed of the progress bar when it came out. I mean, I just opened up Chrome Beta (I think I'm on the Beta channel). I was on Canary for a while. Anyway, I just noticed, literally this morning, that I'm on Chrome 27, and I swear it was 17 really recently. Which brings up this idea of…I guess you really don't think about version numbers anymore. It really is all about feature detection.

Paul Irish: Yep, absolutely. The Chrome way, especially when it comes to users, is there are no version numbers. Chrome just updates. You now have a new feature, and that's the way we talk about it to users. To developers, we do have to be a bit more specific, and that's why sites like CanIUse are really helpful when it comes out of tracking, when Web Audio API, when WebGL, etc., all landed and what version. But, of course, inevitably feature detection is the right way to be handling this in Chrome and in all the various mobile devices that actually are sometimes a bit trickier when it comes to this sort of thing.

Scott Hanselman: Mm hmm. I get the impression (and I don't work for IE nor do I know anyone over there, so this is just me as Web guy observing this) that they seem to believe that things might be a little more slower. It's like you know how some people are on Canary, some people are on Beta, some people are on Stable. How often does Chrome Stable rev?

Paul Irish: About every 12 weeks.

Scott Hanselman: Okay. So people think that Chrome is updating like every 3 or 4 hours. But realistically, it's like 3 or 4 times a year, right? It's only half the speed of Ubuntu, right?

Paul Irish: Oh, I'm sorry. Six weeks is Stable.

Scott Hanselman: Six weeks is Stable? Okay. So, if IE starts doing it every year or every 18 months or something like that, as people on the Web are presuming, the argument is always made that enterprises can't go that fast. They all need IE6 still. Do enterprises use Chrome, and does it revving so fast freak out their IT departments?

Paul Irish: I don't know what the enterprise situation is for Chrome. I do know that when Firefox announced that they had a new release model the enterprise community was really upset, and that led Firefox to introducing the ESR, the Extended Support Release, which I think is equal to Firefox 10. And it's supported for a year. So that was their way of…whereas the rest of Firefox ships every 6 weeks just like Chrome. Chrome has supports for being able to turn off automatic updates, as much as I hate that idea of….There is support. There is MSI Installer, and administrators can turn off updates. But I would be very unhappy if I met someone that did that. The consumer-side Web deals with the moving, the rapidly changing landscape of browsers, and it deals with new browser versions and it feature-detects and all those things. I think the Internet-side web apps are just…there was a time when people didn't know how to develop them very well, and that's why they got locked in to, "it only works in this browser version." But I think that has changed a lot, especially with the changes that IE has made, comparing IE6 to IE9. It's a completely different browser, and anyone developing for the IE9 browser is not going to have a problem when IE10 comes down in their company. We're in a bit of a different landscape, so I think that the concerns of "the web app doesn't work anymore" are mostly behind us.

Scott Hanselman: Yeah. I think that there's…it's all just fud, right. It's all just fear that the expense reporting system we wrote 12 years ago is going to suddenly break, and we don't know where that code is anymore. It sounds silly, but that's a classic problem for an enterprise.

Paul Irish: Yeah, yeah.

Scott Hanselman: Hmmm. So if WebKit is making a move, and we've still got Firefox, but now Opera has announced that they are going to go WebKit, presumably in a fairly orderly fashion, I'm starting to see all these blog posts around what they call WebKit monoculture. So WebKit, some people say it moves the Web forward, which I think we can all agree it kind of did, but, at the same time, people are saying, "Well maybe WebKit will hold us back because then there will only be WebKit."

Paul Irish: Right. I think on desktop this doesn't matter. On desktop, this changed especially as Opera doesn't seem to have as big of an impact. But on mobile, I think that's probably what precipitated a lot of it. It's very hard, I imagine, for Firefox, for IE, and previously for Opera, to get compatibility on the mobile Web because …WebKit has had an early lead there. Web developers using bad practices have…

Scott Hanselman: You mean web developers targeting mobile Safari?

Paul Irish: Yeah, yeah. And they've targeted that side, that flavor, of the Web platform, and in some cases that has not be interoperable. Maybe they're missing vendor prefixes. Maybe they're using things that are only available in iOS WebKit, but it lead to an issue where there's a bad compatibility thing. Mozilla actually did a study and found…they basically went through the top 10,000 sites and looked at the vendor prefixes that they used. Let's say for every site that used webkit-transition, about 15% of those sites also had ms-transition. And probably, just a few bit more would have the unprefix transition which is what IE10 actually uses. There was a big problem that the authoring side didn't readily expect a widely diverse browser engine mobile Web, so that was web developers' fault. So Opera made this move…I mean it's going to be good for the users. A lot of the concern, I think, in the WebKit monoculture conversation is that, when WebKit has some quirk that is unique to WebKit and that is not either in the spec or is not shared by other browsers, and that just becomes the way that everyone expects things to work. So bugs will be filed against other browser vendors saying, "Why don't you treat border radius with elliptical shapes like this?" or something because WebKit does it. I can understand why basically specs need to trump WebKit's behavior, and I think there are a lot of ways to resolve this. Right now, the testing infrastructure between all the browsers could be better. All the browsers have really great tests that they run against, but they're not shared and they're not executed across the vendors as much as they should be. These are fairly interoperable implementations of the same features, but we could do better there.

Scott Hanselman: Yeah.

Paul Irish: That's one of the things that I've been talking about internally inside the Chrome team, but also we had a W3C testing workshop recently around making this setup a lot better, where all the browsers are kind of having congruent feature implementations.

Scott Hanselman: Mm hmm. Why don't specs more trump implementations? I can understand where, if there was only one browser and it was only WebKit, it would be the one that implemented the spec and then that would be the spec. It would become the de facto spec. I get why people would be concerned about that, but why do these things even go wrong? I mean, if you look on Stack Overflow, almost every question on the Web now is "these three browsers do it the way we think the spec works, and this one interpreted it differently." There was some…I forgot what it was. There was some hover pseudo-element thing in IE10 or whatever, and it's like everybody does it this way except they read it that way. Don't they all get in a room and chat about this? How does that even happen?

Paul Irish: They do, yeah. There are a lot of conversations around what behaviors should be, and that gets into the spec. You look at any mailing list, like www.style.com. If you let that into your inbox, you'll be drowned really quickly. But there's a lot of really healthy conversations across browser vendors on what the behavior should be and all these edge cases. It's really fantastic to watch. I think, though, at the same time while there's a debate like that, there is shortage of people who are writing specs and especially people who are authoring test suites, especially early on in spec development. So what that means is that vendors…browsers will then be implementing something and they'll be writing their own tests, but it won't be shared across things. It's like this minor logistical issue that actually is being changed. A lot of the proposals that have come out of Adobe, things like some web components things, those are starting out very early on with cross-vendor test suites. But in other cases, I think like CSS Transforms, did not really have that sort of coverage early on so some of the implementations got a touch different.

Scott Hanselman: Do you think this time next year we're going to start to defragment all of this and get our heads on straight, or are we going to continue to jockey for position like this?

Paul Irish: Yeah, there's a lot of really good things happening in the space right now. W3C actually just added someone full time…his name is Tobie Langel. He's the W3C honcho at Facebook, and he is now full time for one year trying to fix the testing infrastructure so that all vendors are tested across the same stuff and that it's painless for them to maintain that. I'm being kind of cranky about the situation right now, but I have a lot of optimism for how it's going in the right direction.

Scott Hanselman: I wonder what that must feel like. I mean, not to overstate Tobie's position, it's like we have one guy for one year to fix this thing that could change the face of the Web forever.

Paul Irish: (laughter)

Doesn't it seem like somebody might just want to throw in some money to help out? Put a couple of people on this full time and get some things done?

Paul Irish: Yeah. Yes. All browser vendors have people that focus exclusively on testing, but there's a connection point that needs to be improved regarding browsers' own testing and then testing at the standard and the shared level. We're going to see more investment there I expect, and should just make everything a lot better.

Scott Hanselman: I think a lot of people who are making the move, a lot of enterprise developers are making the move, from server-side development where they really work their backend servers to make HTML and ship angle brackets across the wire, are now moving into a balance where there is some work that happens on the server and a lot of work that happens on the client. They find this world where we're all jockeying for position. There's a lot of personality-driven development.

Paul Irish: Yeah!

Scott Hanselman: "You're not using RequireJS. You suck!" "You're not using this, you suck!" It's almost like…with Twitter, the idea of followers, you pick your favorite JavaScript developer or you pick your favorite company, and then you want to use their stuff. It all started with the whole jQuery, MooTools, Mochikit – kind of like a smack down, and then one of them won and now we're fighting about MVC frameworks.

Paul Irish: I think there is something about the web community that really just prizes celebrity – whether it's a project or a person – and it really is all about…I mean, part of it's nice. Part of it's about a strong and loyal community and a community that you have camaraderie with and you learn from. Then the evil side is that sometimes it's just not pushing us in the right direction. I got some feedback on one of my projects a while ago that was along these lines. Somewhere around when HTML5 Boilerplate launched, the copy on the homepage was very…was very showy and it was very like, "This stuff will just make you look cool." (laughter)

Scott Hanselman: Yeah….

Paul Irish: I wrote it in a way that I was so excited about it and I was just excited to share it, but it wasn't explaining what this does for you and how it…It was more getting you excited.

Scott Hanselman: Do you think that's a youthful exuberance? There are people who will come out with some pick-a-noun.JS and then they'll go and make a gorgeous site and it will be like, "such and such and such and such." And then go BOOM, in all caps. Like BOOM? That's our tagline? BOOM? You know what I mean?

Paul Irish: And so, yeah…

Scott Hanselman: It's okay to be excited though.

Paul Irish: Yeah. No. It's okay. One of the things that I…it's been really enlightening to see the flex developer community kind of entering into JavaScript and HTML5 because they've come from a completely different background. They're used to a platform that offers a lot of things, they kind of have to put together on their own. And they come and they see these projects, and they're just like, "Why are these projects marketing themselves so hard at you? Why aren't they just telling me bullet by bullet the feature set that they offer?" And, I understand. Part of it is the web community doesn't have the vocabulary established that people like the flex community have. The other part is that…I agree that things could be a little bit more…just plain, and readable, when it comes to what exactly you're getting out of this project and how you're going to want to partner your use of this, of this tool or project, with other things.

Scott Hanselman: Yeah. I think that it feels like that there are small companies, small startups, 5 or 10 people in a room, really excited, full of energy and caffeine, head in one direction, and then there's kind of this tired, more enterprise-y developer who is just like, "just solve this problem for me. Do it in a reliable way. Don't disappear." So there's this pragmatism aspect of things, almost like there's a triangle. There's pragmatism; there's showy and flashy; and then there's purist – is this code haiku quality.

Paul Irish: Right, right.

Scott Hanselman: And there's this constant balance between…take any library and you could plot it on this triangle as far as "is it flashy?" "Is it showy?" "Is it trying to be LISP like?" "Is it trying to be CoffeeScript where it's gorgeous and *look what we did in 12 lines of code*?" Or is it that there's no library, it's just really a gorgeous front end and there's nothing real deep there?

Paul Irish: Right. It seems like you want…when you plot it out like that in a triangle, I feel like I want something that's kind of in the middle. I want something that is eloquently written, and I want it to sell itself well enough that I know there's going to be a community that's around it and can support it and I can learn from.

Scott Hanselman: Yeah.

Paul Irish: But I also want to know that its engineering principles are good and that the problem that it's trying to solve is large enough for me to sink some time into learning it.

Scott Hanselman: Exactly. I think…I'm looking here online. I thought there was a project that just came out. I want to say it was called Low Bar.

Paul Irish: Slowmo.JS? No.

Scott Hanselman: No. Shoot. What was her name? Aw, I'll find it. But there was a project that just came out that was trying to be like Underscore but it was an Underscore competitor?

Paul Irish: Oh. Lo-Dash.

Scott Hanselman: Dash. Lo-Dash.

Paul Irish: Yeah, Lo-Dash, written by John-David Dalton…he's actually a JavaScript Performance Manager at Microsoft. This is his open source project. He came out with this…originally, it started as a fork of Underscore because John tried to get some changes into Underscore that the community didn't allow in. He created a fork. He has now been allowed back onto the project as a contributor and collaborator, and so he is contributing to both his fork and alternative and the original project which is an interesting situation.

Scott Hanselman: Right.

Paul Irish: But it's also…the Lo-Dash community, while originally it was like this outcast and everyone was "I don't know if I can trust this," now it's actually a very well-celebrated and shared project.

Scott Hanselman: Exactly. And I bring that up as an example where Kit Cambridge's article that said – *Say "Hello" to Lo-Dash* - was presented in a way that was practical, pragmatic, flashy – side-by-side demos – but extremely like, "All right. Here's what we have to offer." So I look at that, in my mind, the one that I've thought of in the last few months that's right in the middle of that triangle.

Paul Irish: Yeah. It's a really solid project. They've done a great job there.

Scott Hanselman: Yeah. It's pretty exciting. At a lower level, I feel like we're reinventing a lot of work. "All right. I'm going to make the world's great to-do application." First, I need to decide how I'm going to manage async. First, I'm going to decide how I do module loading.

Paul Irish: Sure.

Scott Hanselman: It seems like we're in a very scary time of flux right now. If I was going to do a startup and do a lot of JavaScript today, I might find half of my project being thrown out when ES6 comes out next year, or whenever things happen. You know what I mean?

Paul Irish: Yeah. There's a lot of things going on right now. Another part is "I want to build a web app." There's a lot of parts that are going to…the best way to build a large app is to never build a large app and to build a collection of small modules. Things that work together.

Scott Hanselman: Mm hmm.

Paul Irish: How am I adding that into my page? Twenty-five script tags? Probably not. I should be using modules. These days, people go the route of authoring in AMD modules or will use Common.JS modules and use something like node-browserify. They both have their advantages and disadvantages, but I think that nowadays it's pretty much impossible to be authoring something of significance and not be authoring your code in modules. I'm excited about what ECMAScript 6 does because modules are…right now, we have these hacks to make this work, to protect the global scope and things like that. ECMAScript 6 offers a real native way to do it.

Scott Hanselman: Mm hmm.

Paul Irish: But also there's a lot of other pieces coming around this. One is…when it comes to building an app, I'm going to have my application code. I'm going to split it up. But I'm also going to have dependencies on third-party code. And right now, let's say I want to use Backbone. So I open my browser and I Google Backbone. I download it. I move it from my Downloads folder into my Application folder, and I add it into my page somehow. When I update it, I go through a similar thing. Like I'm watching Twitter to find out when it updates and then I go…These manual steps – oh, I didn't even mention the fact that it has dependencies of its own that I need to go find on the Web, too. I feel that we're at this point where we, that package management could really solve a lot of problems for us. Make us more productive. Keep our libraries up to date. Help us out a lot with that.

Scott Hanselman: Isn't there a lot of…isn't it multifaceted, though? I mean, there's bringing it down. There's putting it somewhere. There's version management, dependency management like you mentioned. There's like the RPM, NPM, NuGet, GEMS side of things. Development time.

Paul Irish: Right.

Scott Hanselman: Then there's runtime loading and management of that.

Paul Irish: Yep.

Scott Hanselman: But then bundling and minification.

Paul Irish: Yeah, the optimization side, on that side. So a good solution has to handle all these things, and a few people have tried different approaches. There was a project a bit ago called BPM, Browser Package Manager, and it kind of tried to tackle all three pieces. So it not only tracked the retrieving of packages but also had its own preview server and then handled minification itself as well. Other projects nowadays…I'm involved in a project called Bower that was originally started by Twitter but now is a very large development, open source community that is supporting it. It is pretty much only just…there's a registry and there's a way to pull things down from it, resolve the dependencies, get those, and then keep things up to date. And there's hooks to connect to these other parts of runtime module loading and build time optimization, but I think that, without basically adopting package management solutions, we're holding ourselves back.

Scott Hanselman: Mm hmm.

Paul Irish: I see people remaining at the same level of the Web platform. They're just only building so high. People get afraid of having dependencies, of using third-party code, and I think it's really holding back the sophistication of what we can actually develop on the Web platform. I'm just really excited to see more adoption, more tooling, around things, you know? I don't want to have to manage my NuGet and my NPM and my other client-side package management all uniquely if I don't have to, and I think there are some opportunities for integrations but also plenty of other development. I'm pretty active in the developer community there and invite anyone else to join us.

Scott Hanselman: One of the things I struggled with a lot yesterday, that I guess FlexBox will fix at some point – I was messing with some CSS and I had three browsers open on three monitors. I was doing a bunch of breakpoint responsive stuff, and I was chasing boxes around. I basically had a grid of boxes and I wanted them to resize appropriately, and I was deciding whether I wanted to go and use – what is the one, the name of bricks. It's the automatic resizing – Masonry!

Paul Irish: Masonry, yeah.

Scott Hanselman: So I was thinking, "Well, maybe it's time to just rip this whole design out and do Masonry." And I opened up Chrome Tools, and I get the CSS perfect. Then I bumped something, and it was back the way it was again. And I lost everything. I said to myself, "How come I'm not typing in here and having it automatically go back?" So I go and look around on the Web, and there's a half dozen different projects out there where they're trying to close that browser tools loop to say, "I'm in a mode now where I've modified my CSS and want to push it back to the server now." That should be a spec, you see what I'm saying? Every browser should support that.

Paul Irish: Yeah. There's a few ways, a least in the Chrome Dev Tools, to tackle that. One is command-Z works these days.

Scott Hanselman: Just undo my…I think I reloaded the page or something.

Paul Irish: That was not too long ago, but it's kind of a secret and a probably undocumented feature. Yeah, that happens. Inside Chrome Dev Tools, we've done a lot of things to create that kind of authoring experience where you can just basically develop inside the Chrome Dev Tools and not even head back to your editor, your IDE. Remy Sharp has a great screencast on this. But you can basically…one of the great things that actually just finished landing…So let's say you're developing your CSS with Sass. There's a port for Sass Source Maps. So I can author in Sass, I can compile it. It comes out in CSS. I see the styles of my DOM as regular, but then I can CTRL-click back to my original Sass while being in the Chrome Dev Tools, make changes right there inside the dev tools, hit save, and then my styles will auto-refresh with the new compiled styles. That's a really cool workflow that you don't even have to leave the dev tools for, and so we're seeing a lot of improvements there. Let's take the workflow that you're doing with all these tools and just separate it and deliver it all to you in one place.

Scott Hanselman: Is the Chrome Dev Tools then opening the file off disk or is there is a listener that I have to implement in my server.

Paul Irish: It is…the compilation step is up to you but, when the compiled file changes, the Chrome Dev Tools picks that up on its own.

Scott Hanselman: Very cool. One of the other things I wanted to talk to you about…one of the last things I wanted to talk to you about is that it's clear that a lot of front-end development is Mac focused and Node focused, and NPM is becoming the package manager for everything. Things like Bower and Yeoman and all those things are all focused at the command line. And the Windows guys are really struggling. If I wanted to get Yeoman running on Windows, it's just going to always be a struggle.

Paul Irish: Mm hmm.

Scott Hanselman: What can…should Microsoft try to help out and make things like Yeoman and Bower and these different things easier to use for Windows developers?

Paul Irish: Hmmm.

Scott Hanselman: You know what I mean. Like Azure, for example. We have the Azure tools in NPM, so you just go NPM, install Azure, and then you go and rock it at the command line. It's cool. But trying to get something line Yeoman working, or Auto-reload, it's always like "for Linux," "for Mac," and Windows in beta." We're trying to figure out do we need Cygwin or not. Ultimately, are we being held back by the fact that we just don't have Bash?

Paul Irish: (laughter) I think the answer to that last question is probably yes. I mean….

Scott Hanselman: Hey, man. It's an honest question. I'm trying…There's a lot of Windows developers out there that want to be on the cutting edge.

Paul Irish: I want to type LS and have it work.

Scott Hanselman: It does work in PowerShell, but, yes, you're absolutely right.

Paul Irish: Okay. Cool, cool. But I did a survey not too long ago about what, of the Chrome Developer Tools audience, what OS everyone was on. And it's 40/40/20, Win/Mac/Linux.

Scott Hanselman: Really?

Paul Irish: Yeah. I was surprised by how high Linux was so I went and double-checked. And, yeah!

Scott Hanselman: I still find that hard to believe. That's pretty crazy.

Paul Irish: Yeah. It's really high! But that makes me make sure that Linux support is a big priority. As for Windows priority, or Windows support for dev tools, I think it's mostly just the history there which is that Windows has always offered kind of gooey developer tools whereas on Mac and Linux it has been more command-line focused. I don't know what developers using Windows prefer, but certainly from a tooling standpoint it's much easier to support command-line tools on Windows than it is to have to build a GUI around it. As long as you're okay with that, that being your audience, someone who's comfortable at the command line.

Scott Hanselman: Yeah, yeah. We're finding that, like in ASP.NET, for example, it's about 60/40 or 70/30 who really like the GUI, but they want a command line, usually in Visual Studio. For example, in VS, there's a Package Console, just like in Sublime. You can bring up the Quake Console, and then do your thing and then throw it away. We use…the NuGet Package Manager has PowerShell inside VS, and inevitably what happens is we'll covert stuff from Bash to PowerShell and then wish we had Bash in Visual Studio.

Paul Irish: (laughter) Yeah.

Scott Hanselman: I want to go into Visual Studio, go and say NPM install, Bower, or Add Yeoman, and then go back into VS. But I haven't really left, you know what I mean?

Paul Irish: Yeah, yeah.

Scott Hanselman: Kind of like VS as EMX.

Paul Irish: Whoa. Whoa.

Scott Hanselman: I want it paneled. I live in VS. I like things about VS, but I want to have a command line that's docked and move smoothly between the two. And trying to figure out – and I just used Yeoman as the example because Yeoman is one of those things that brings in a bunch of other tools. I don't know whether Yeoman will win the Web, but that's just an example of a perspective that…I can't even offer, without busting out my MacBook, I can't even offer ideas on what I think about it.

Paul Irish: On Yeoman, we just underwent a really big refactor and it kind of changed the entire way it works.

Scott Hanselman: Thereby breaking it on Windows.

Paul Irish: Well, no.

Scott Hanselman: The beta does not currently work on Windows.

Paul Irish: So, yeah, I guess so.

(laughter)

Paul Irish: There used to be Yeoman Commands on the command line, and that is now dead and deceased. So a big change in how everything operates. But the 1.0 final, which we expect to ship in the next few weeks (?), will have Windows support.

Scott Hanselman: I think that NPM and Node and JavaScript executables may be the answer, isn't it?

Paul Irish: I'd be pretty happy with that.

Scott Hanselman: Maybe I can't go and go all 'SED' and 'AWK' my way to glory, but if I can type Yo-this and Yo-that, and it's just a Node app – that would be great. That's fantastic.

Paul Irish: Yeah, absolutely. The environment of NPM running so well on Windows these days, I think, was a really big boost for both sides. I've really liked seeing the tool ecosystem written in Node kind of taking off. It's not just for modules that are used in an app but are now the build tools that you're using to construct those apps. I'm really excited about that move.

Scott Hanselman: Yeah, definitely. Cool. Well, thanks so much for talking to me today. I really appreciate it.

Paul Irish: Absolutely. It's been great.

Scott Hanselman: Awesome. Well, Paul Irish…you can check him out. You can Google for Paul Irish and you'll find all sorts of great stuff, his YouTube channel, his blog, an HTML5 Boilerplate.

This has been another episode of Hanselminutes. We'll see you again next week.